帧率是最重要的标准

确定性能瓶颈：

1，渲染

1，CPU

确定：

Time ms列排序，渲染方法排在前面

优化：

Reducing the number of objects being rendered

Reducing visible objects

Camera’s Far Clip Plane property

Camera’s Layer Cull Distances

Occlusion culling

Reducing the number of times each object must be rendered

渲染路径，

光照，光照探针

阴影

Combining objects into fewer batches

Static batching

Dynamic batching

Manually combine meshes

GPU instancing

Texture atlasing

2，GPU

确定：

支持GPU Profiler：

查看GPU和CPU消耗的时间

不支持GPU Profiler：

查看Gfx.WaitForPresent消耗时间

优化：

Fill rate

Check：降低分辨率

Fix the problem：

Complex fragment shaders

Overdraw

Image effects

Memory bandwidth

Check：降低图片质量和Quality中的质量

Fix the problem：

Texture compression

Mipmaps

Vertex processing

Check：

Fix the problem：

LOD

Normal mapping

Create meshes with a lower vertex count in our 3D art program.

Optimize Vertex shaders

2，GC

确定：

Time ms列排序，GC.Collect()方法排在前面

优化：

Reducing the amount of garbage created

Caching

Don’t allocate in functions that are called frequently

Clearing collections

Object pooling

Common causes of unnecessary heap allocations

Strings

Removing calls to Debug.Log()

Unity Methods that return arrays

Boxing

Coroutines

Foreach loops

Function references

LINQ and Regular Expressions

Manually forcing garbage collection

System.GC.Collect();

3，物理

确定：

Time ms列排序，Physics方法排在前面

4，脚本

确定：

Time ms列排序

优化：

Writing efficient code

Calculating a square root takes a CPU more time to execute than multiplying two numbers.

Move code out of loops when possible

Consider whether code must run every frame

Only run code when things change

Use caching

Use the right data structure

Minimize the impact of garbage collection

Use object pooling

每当在Animator、Material或Shader上使用Set或Get方法时，请使用什么方法而非字符串值方法？

Avoiding expensive calls to the Unity API

SendMessage() and BroadcastMessage()

Find()

Remove Empty Update()

Vector2 and Vector3

Camera.main

5，内存

资源审核

通用资源规则：对于纹理有哪些？

1，禁用 read/write enabled 标志 2，尽可能禁用 Mipmap 3，压缩所有纹理 4，实施合理的纹理大小限制

通用资源规则：对于模型有哪些？

1，禁用 Read/Write enabled 标志；2，在非角色模型上禁用骨架；3，在动画模型上启用Optimize Game Objects选项；4，尽可能使用网格压缩；5，注意网格渲染器设置

通用资源规则：对于音频有哪些？

1，适合平台的压缩设置 2，将音频剪辑强制设置为单声道 3，降低音频比特率 4，背景音乐LoadType设置为Streaming

内存纹理重复的识别和AssetBundle的卸载问题

许多Unity项目在每帧都有几十或几百 KB 的临时数据分配给什么？这种情况通常对项目的性能极为不利？

强烈建议所有开发人员在项目处于什么状态时最大限度减少托管堆内存分配。非交互操作（例如场景加载）期间的内存分配很少产生问题？

6，UI

Fill-rate：

Eliminating invisible UI

Disabling invisible camera output

UI shaders and low-spec devices

UI Canvas rebuilds

Child order

Intermediate layer

Splitting Canvases

On one Canvas, place all elements that are static and unchanging

On the second Canvas, place all of the “dynamic” elements

Input and raycasting in Unity UI

Only enable the ‘Raycast Target’ setting that must receive pointer events.